Operating System/2™ Version 1.2 Programming Tools and Information

Technical Update

70F4539

Operating System/2™
Version 1.2
Programming Tools
and Information
Technical Update

First Edition (November 1989)

The following paragraph does not apply to the United Kingdom or any country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time.

It is possible that this publication may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or services in your country.

Any reference to an IBM licensed program or other IBM product in this publication is not intended to state or imply that only IBM's program or other product may be used.

Requests for technical information about IBM products should be made to your IBM Authorized Dealer or your IBM Marketing Representative.

©International Business Machines Corporation 1989. All Rights Reserved.

Special Notices

The following names, used in this publication, are trademarks or registered trademarks of International Business Machines Corporation in the United States and/or other countries:

IBM
IBM C/2
IBM COBOL/2
IBM FORTRAN/2
IBM Macro Assembler/2
Operating System/2
OS/2
Personal System/2
PS/2
Presentation Manager

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Commercial Relations, IBM Corporation, Armonk, NY 10504.

About This Book

This *Technical Update* provides additional programming information to the OS/2 Version 1.2 Programming Tools and Information library. This book describes new features that have been added to the product and amends some of the information that was published with the product. Contents of the READ.ME file in the \TOOLKT12 directory of the Programming Tools and Information package have also been incorporated into this book.

Who Should Read This Book

This book is for application designers and programmers who are using the publications supplied as part of the OS/2 Version 1.2 Programming Tools and Information.

You should understand the OS/2 services that are available to the users of the operating system. These services are described in the publications that are shown in the product library diagram at the front of each of the books.

How to Use This Book

This book contains a section for each of the following books:

Book Title	***Book Number***
Programming Guide	64F0273
Building Programs	64F0274
Control Program Programming Reference	64F0275
Presentation Manager Programming Reference: Volume 1	64F0276
Presentation Manager Programming Reference: Volume 2	64F0277
Presentation Manager C/2 Bindings Reference	64F0278
Presentation Manager Macro Assembler/2 Bindings Reference	64F0279
Presentation Manager COBOL/2 Bindings Reference	64F0280
Presentation Manager FORTRAN/2 Bindings Reference	64F0281
I/O Subsystems and Device Support: Volume 1	64F0282
I/O Subsystems and Device Support: Volume 2	64F0283

Each section refers to pages in these books, and the items are in page order. You may want to write a note on the relevant pages of the original books to warn you that there is extra information contained in this book.

Contents

Programming Guide 1
DLG1 and DLG2 Nonreentrant Sample Applications 1
The Heap Manager 1
PMPRINT Queue-Processor Parameters 1
Graphics Attributes 1
Alphanumeric Video Output 3
 Setting the Display Mode for Text-Windowed and Full-Screen Applications 3
File and Disk Management 3
 Controlling Access to EAs 3
 Defining EA data structures 4
Limits for Various Operating System Resources 5

Building Programs 7
LINK Options 7
 Summary of LINK Options 7
 Option Descriptions 7
/? Viewing the Options List 7
/BATCH Run in Batch Mode 7
/NOLOGO Disable Sign-On Banner 8
/NONULLSDOSSEG Order Segments Without NULLs 8
/PMTYPE Name Application Type 9
/TINY Generate .COM File 9

Control Program Programming Reference 11
DosFindFirst2, DosMkDir2, DosOpen2, DosQFileInfo, DosQPathInfo, DosSetFileInfo, and DosSetPathInfo 11
DosShutdown 11
VioSetMode 11

Presentation Manager Programming Reference: Volume 1 13
Data Types 13
 BITMAPINFO 13
 BITMAPINFOHEADER 13
 MARGSTRUCT 13
 MLE_SEARCHDATA 13
 OVERFLOW 14
Graphics Function Calls 14
 GpiCreateLogColorTable 14
 GpiQueryBitmapParameters 14
 GpiQueryLogColorTable 14
 GpiSetClipRegion 14
 GpiSetMarkerSet 15
 GpiSetMetaFileBits 15
 GpiSetPatternSet 15
Advanced Video Function Calls 15
 VioQueryFonts 15
Window Function Calls 15
 RECT Structure 15
 WinCreateHeap 16
 WinCreateStdWindow 16
 WinInitialize 16
 WinLoadDlg 16
 WinRegisterClass 16

WinSetClassMsgInterest . . . 16
WinSetMsgInterest . . . 17
WinTrackRect . . . 17

Presentation Manager Programming Reference: Volume 2 . . . 21
Button Control Window Processing . . . 21
Frame Control Window Processing . . . 21
List Box Control Window Processing . . . 21
Multi-Line Entry Field Control Window Processing . . . 21
WM_CONTROL . . . 21
MLM_FORMAT . . . 22
MLM_QUERYSEL . . . 22
MLM_SEARCH . . . 22
MLM_SETFORMATRECT . . . 22
Prompted Entry Field Control Window Processing . . . 22
Help Manager Messages . . . 22
HM_ACTION_BAR_COMMAND . . . 22
Resource Files . . . 22
Code Pages . . . 23
Standard Bit-Map Formats . . . 24
Bit-Map File Format . . . 24

Presentation Manager C/2 Bindings Reference . . . 27
Data Types . . . 27
HLIB . . . 27
MARGSTRUCT . . . 27
OVERFLOW . . . 27
MLE_SEARCHDATA . . . 27

Presentation Manager Macro Assembler/2 Bindings Reference . . . 29
Data Types . . . 29
MARGSTRUCT . . . 29
OVERFLOW . . . 29
Window Function Calls . . . 29

Presentation Manager COBOL/2 Bindings Reference . . . 31
Introduction . . . 31
Named Constants . . . 31
Data Types . . . 31
Ranges . . . 31
MARGSTRUCT . . . 31
OVERFLOW . . . 31
Window Function Calls . . . 31
WinQuerySwitchList (WMQSLI) . . . 32

Presentation Manager FORTRAN/2 Bindings Reference . . . 33
Introduction . . . 33
Named Constants . . . 33
Data Types . . . 33
HPROGARRAY . . . 33
MARGSTRUCT . . . 33
OVERFLOW . . . 33
Profile Function Calls . . . 34
PrfQueryProgramHandle (PRQPHD) . . . 34
Window Function Calls . . . 34
WinQuerySwitchList (WMQSLI) . . . 34

I/O Subsystems and Device Support Volume 1: Device Drivers 35
Device Driver Overview 35
Device Driver INIT-Time Function Call Summary 35
Device Driver Services 35
DevHlp Services and Function Codes 35
DevHlp Service Modes 35
DispMsg Display Message 35
VideoPause Suspend/Resume Video Active Threads 36
Device Drivers 36
Keystroke Monitors 37
Monitoring Character Device Data Streams 37
Device Monitor Support Limitations 37

I/O Subsystems and Device Support Volume 2: Presentation Driver Interface 39
GreQueryLogColorTable 39
GreCreateLogColorTable 39
GreEscape, DEVESC_GETCP 39
GreEscape, DEVESC_SETMODE 39
GreEscape, DEVESC_DRAFTMODE 40
GreDeviceSetAVIOFont 40
GreDeviceSetDCOrigin 41
GreSetupDC 41
SplQpQueryDt 41

Additional Information 43
Using the MODE MONO Command 43
Online OS/2 DM and DTL Reference 43
Online OS/2 Programming Reference 43

Programming Guide

DLG1 and DLG2 Nonreentrant Sample Applications

On page 11-1, change the file name DLG1COB.COB to DLG1COB.CBL.

On page 12-1, change the file name DLG2COB.COB to DLG2COB.CBL.

The Heap Manager

On page 13-22, under "A heap has the following properties:", change the third item to:

- All pointers to memory objects within a heap are 16-bit offsets from the start of that segment. All memory objects are aligned on a ULONG boundary.

PMPRINT Queue-Processor Parameters

On pages 37-7 and 37-8, make the following changes to "Passing Parameters to the PMPRINT Queue Processor":

1. Add this queue-processor parameter:

 CDP=*nnn*

 The CDP parameter changes the code page to be used for printing alphanumeric data. *nnn* must be a valid code-page identifier; for example, the identifier for the multilingual code page is 850. You can use this parameter with all spool-file formats.

 The default for this parameter is to leave the code page unchanged.

2. Delete all information about the MAP parameter. This parameter is not supported.

Graphics Attributes

The following information replaces that given on page C-1. Use it when you read these sections:

- Page 19-3, The Current Position
- Page 20-1, Graphics Primitives and Their Attributes
- Page 22-13, Correlating on Retained Graphics
- Page 25-2, Model Space
- Page 26-1, The Clipping Path
- Page 26-2, The Viewing Window.

Every graphics presentation space has a set of *graphics attributes*. A normal presentation space has a larger set of graphics attributes than a micro or cached micro presentation space. All these attributes have default values, so they will all affect your graphics, even if you have not explicitly set them.

The graphics attributes can be divided into two groups:

Group One Attributes that form part of the picture and that can vary as the picture is drawn

Group Two Attributes that define the environment in which the picture is drawn.

Group-One Graphics Attributes

The graphics attributes in group one are:

- The current position
- All primitive attributes (including arc parameters)
- The primitive tag
- The model and segment transformations
- The clipping path
- The viewing window
- The segment attributes
- The viewing transformation.

When you draw a segment, the *fast-chaining* segment attribute determines what happens to the current values of the first six group-one attributes. When the fast-chaining segment attribute is set ON (this is the default setting), and you want to use a primitive that depends on an attribute, you must either:

- Set the attribute explicitly for each segment, or
- Leave the attribute at its default setting in the preceding segments.

When you edit or create a segment, you can set the fast-chaining segment attribute off for the first segment (using the GpiSetInitialSegmentAttrs (GSSATI) call) to ensure that each graphics attribute has its default setting. You can set fast-chaining on for subsequent segments.

The values of group-one attributes cannot be guaranteed (except in the situation described below) following the use of:

- Any GpiDraw call
- Any GpiCorrelate call
- GpiCloseSegment (GSSCLS).

If any of these calls is followed by primitives outside a segment, you should explicitly set the required graphics attributes. However, these attributes *are* in the correct state to continue drawing if the last segment drawn is already open when the call is issued and no dynamic segments have been drawn.

The group-one attributes take effect when graphics are sent to an output device, *not* when graphics are defined. For this reason, the GpiQuery calls that retrieve the current values of these attributes are not valid in retain mode. The majority of the calls that set the group-one attributes cause graphics orders to be added to the current segment.

Group-Two Graphics Attributes

The group-two graphics attributes do not normally vary as the picture is drawn, but they have an overall effect on the result of any drawing or correlation operation. This group includes:

- The default viewing transformation
- The page viewport
- The graphics field
- The clipping region
- The pick-aperture size
- The drawing controls.

The GpiQuery calls that retrieve the current values of these attributes are valid in all drawing modes. None of the calls that set group-two graphics attributes causes graphics orders to be added to the current segment.

List of Graphics Attributes

The table on page C-2 lists all the graphics attributes, identifies the GPI call or calls that you use to change the current settings, and lists the default value for each attribute. These default values are the initial settings. The table also shows to which group the attribute belongs.

Alphanumeric Video Output

Setting the Display Mode for Text-Windowed and Full-Screen Applications

On page 30-7, replace the last paragraph on the page with the following:

To set the mode, initialize the data structure and issue **VioSetMode**. VioSetMode initializes the current settings for cursor position and type; however, it does not clear the screen. (In DOS mode only, VioSetMode clears the screen.) To clear the screen, you must issue one of the VioScrollXX calls. See "Scrolling the Screen Image." To reset the cursor position and type, see "Manipulating the Cursor" on page 30-6.

File and Disk Management

Controlling Access to EAs

On page 33-23, replace the second paragraph under **Controlling Access to EAs** with the following:

In addition, operations on EAs are not atomic: any one query or set operation may not complete before any other query or set operation can be performed on the same object. For example, if an error occurs before an entire list of EAs has been set, all, some, or none of them may actually remain set on the file object. This means that EAs may not remain in a consistent state unless the order in which the operations are performed can be guaranteed.

Defining EA data structures

On page 33-25, replace **Figure 33-8 EA data structures** with the following figure.

```
struct FEA
       unsigned char    flags;          /* byte of flags                */
       unsigned char    cbName;         /* length of EA name            */
       unsigned short   cbValue;        /* length of EA value           */
       unsigned char    szName[];       /* ASCII EA name                */
       unsigned char    aValue[];       /* free-format EA value         */
       ;

struct FEAList
       unsigned long    cbList;         /* length of list               */
       struct FEA list[];               /* set of FEAs                  */
       ;

struct GEA
       unsigned char    cbName;         /* length of EA name            */
       unsigned char    szName[];       /* ASCII EA name                */
       ;

struct GEAList
       unsigned long    cbList;         /* length of list               */
       struct GEA list[];               /* set of GEAs                  */
       ;

struct EAOP
       struct GEAList far * fpGEAList;  /* set of GEAs                  */
       struct FEAList far * fpFEAList;  /* set of FEAs                  */
       unsigned long  offError;         /* offset of FEA err            */
       ;
```

Limits for Various Operating System Resources

Appendix G contains tables of system limits. The following replaces the table on page G-3.

Resource	System Limit	Available to Applications
Processes	511	504
Threads	511	483
Threads per process	53	53
Open file handles	64KB	—
Pipes	Part of open file handles	Part of open file handles
Pipe size	64KB	—
Queues	1,448	1,256
Queue priorities	16	—
Queue elements per queue	3,268	—
System semaphores	256	213
Semaphores with outstanding waits	128	126
Maximum semaphores per DosMuxSemWait	16	—
System signals per process	3	—
User signals per process	3	—

Threads:

The minimum available space on the stack for a thread calling OS/2 functions is 4KB. The system allows only one timer per thread.

System Semaphores:

System semaphores have a *use* count and a *reference* count:

- The *use* count is the number of events a system semaphore can represent. A system semaphore is limited to having no more than 255 concurrent uses. A *use* is any one of the following events:
 - Creating a system semaphore.
 - Opening a system semaphore.
 - Specifying the handle of the system semaphore on a call to an OS/2 API, for example, DosTimerAsync.
- The *reference* count is the number of threads waiting for an exclusive system semaphore. A system semaphore is limited to having no more than 255 concurrent references.

Building Programs

LINK Options

Summary of LINK Options

On pages 7-1 and 7-2, add the following LINK options:

Option	*Description*
/?	Writes a list of the available options to the screen.
/BATCH	Disables default prompt requesting new path name.
/NOLOGO	Disables the sign-on banner displayed when LINK starts.
/NONULLSDOSSEG	Forces ordering of segments without NULLs.
/PMTYPE	Specifies the type of application being generated.
/TINY	Generates a .COM output file rather than a .EXE file.

On page 7-1, replace **NOEXTENDEDDICTIONARYSEARCH** with the following:

Option	*Description*
/NOEXTDICTIONARY	Prevents LINK from searching the extended dictionary.

Option Descriptions

A description of the individual options follows.

/?
Viewing the Options List

Purpose

To write a list of the available options to the screen.

Format

```
/?
```

Remarks

- /? option and /HELP option provide the same information.
- Do not give a file name when using the /? option.

/BATCH
Run in Batch Mode

Purpose

To disable default prompt generated by LINK. LINK, by default, prompts you for a new path name whenever it cannot find an object file or library it was directed to use. Instead, LINK generates an appropriate error or warning message and leaves the external reference unresolved.

Format

/BATCH

The minimum abbreviation is **/BA**.

Remarks

The primary use of this option is to call LINK from a batch file or a *Make* utility.

Note that this option does not affect prompts for command line input.

/NOLOGO
Disable Sign-On Banner

Purpose

To disable the sign-on banner displayed when LINK starts.

Format

/NOLOGO

The minimum abbreviation is **/NOL**.

/NONULLSDOSSEG
Order Segments Without NULLs

Purpose

To arrange segments in a certain sequence. This option is the same as the /DOSSEG option, except that /NONULLSDOSSEG does not insert 16 bytes of NULLs in front of the segment named _TEXT (if the _TEXT segment is defined).

Format

/NONULLSDOSSEG

The minimum abbreviation is **/NON**.

Remarks

Linking with the standard C-2 run-time libraries automatically enables the /DOSSEG option in the startup module. Use the /NONULLSDOSSEG option to override the /DOSSEG comment record.

/PMTYPE
Name Application Type

Purpose

To specify the type of application being linked.

Format

/PMTYPE[*:type*]

The minimum abbreviation is **/PM**.

Restrictions

This option is valid for OS/2 executable files only.

Remarks

The /PMTYPE option is equivalent to the NAME statement in a module definition file. A keyword in <type> is the same as a keyword in a NAME statement, as demonstrated by the following:

<type> field keyword	**Equivalent keyword in NAME statement**
PM	WINDOWAPI
VIO	WINDOWCOMPAT
NOVIO	NOTWINDOWCOMPAT

/TINY
Generate .COM File

Purpose

To create a file that defaults to a .COM file extension rather than a .EXE file extension.

Format

/TINY

The minimum abbreviation is **/T**.

Restrictions

- Only one physical segment is permitted with each program. You can declare more than one segment in an assembly-language program, but the segments must be in the same group.
- Far-segment references are not permitted in the program. This means segment addresses cannot be used as immediate data for instructions. For example, the following instruction is not allowed:

```
mov     ax,CODESEG
```

- Programs for OS/2 cannot be linked to the .COM file. .COM files are valid only in real mode.

Remarks

- Using this option, LINK automatically sets the /FARCALLTRANSLATION option to optimize far calls. This increases your chances of generating a .COM file format.

Note: LINK creates a separate file with debugger information when the /CODEVIEW option is used with the /TINY option. LINK supplies the same filename as the base filename but with a .DBG extension.

Control Program Programming Reference

DosFindFirst2, DosMkDir2, DosOpen2, DosQFileInfo, DosQPathInfo, DosSetFileInfo, and DosSetPathInfo

In each of the following calls:

DosFindFirst2	Page 2-93
DosMkDir2	Page 2-183
DosOpen2	Page 2-209 and Page 2-210
DosQFileInfo	Page 2-251
DosQPathInfo	Page 2-272
DosSetFileInfo	Page 2-327
DosSetPathInfo	Page 2-341

change the information in the **Flags** definition under **list (FEA)** to:

Bit	Description
7	Critical EA.
6-0	Reserved and must be set to zero.

If bit 7 is set to 1, this indicates a critical EA. If bit 7 is 0, this means the EA is noncritical; that is, the EA is not essential to the intended use by an application of the file with which it is associated.

DosShutdown

On page 2-360, at the top of the page, change the wording in the box to read:

DosShutdown (Reserved)

VioSetMode

On page 5-72, add the following at the bottom of the page:

Family API Considerations

Some options operate differently in the DOS mode than in the OS/2 mode. Therefore, the following consideration applies to VioSetMode when coding for the DOS mode:

VioSetMode clears the screen.

Presentation Manager Programming Reference: Volume 1

Data Types

BITMAPINFO

On page 2-2, change the description of the **length** field to:

length
Length of structure.

The value of this depends on the particular language in use.

BITMAPINFOHEADER

On page 2-3, change the description of the **length** field to:

length
Length of structure.

The value of this depends on the particular language in use.

MARGSTRUCT

On page 2-21, this data type is now named *MLEMARGSTRUCT*.

Change the constants listed under the **flags** parameter as follows:

MLF_MARGINLEFT should be MLFMARGIN_LEFT
MLF_MARGINRIGHT should be MLFMARGIN_RIGHT
MLF_MARGINTOP should be MLFMARGIN_TOP
MLF_MARGINBOTTOM should be MLFMARGIN_BOTTOM.

MLE_SEARCHDATA

On page 2-23, this data type should read:

size *(COUNT2B)*
Size of *MLE_SEARCHDATA* structure.

findstring *(STR)*
String to search for.

replacestring *(STR)*
String to replace with.

findstringlength *(SHORT)*
Length of **findstring** string.

replacestringlength *(SHORT)*
Length of **replacestring** string.

start *(IPT)*
Point at which to start search, or point where string was found.

non-negative Point at which to start search

negative Start search from current cursor location.

stop *(IPT)*
Point at which to stop search.

non-negative Point at which to stop search
negative Stop search at end of text.

foundstringlength *(SHORT)*
Length of string found at **start**.

OVERFLOW

On page 2-24, this data type is now named *MLEOVERFLOW*.

Graphics Function Calls

GpiCreateLogColorTable

On page 4-50, add the following to LCOLF_INDRGB in the **Format** parameter:

Each index specified must be greater than or equal to zero.

Add the following to the **Start** parameter:

The starting index must be greater than or equal to zero.

GpiQueryBitmapParameters

On page 4-138, change the **Data** parameter to input/output.

Add this programming note:

The **length** field of the *BITMAPINFOHEADER* structure can be set by the application before making this call, but this is not mandatory.

This note replaces the one described in the READ.ME file in the OS/2 Version 1.2 Programming Tools.

GpiQueryLogColorTable

On page 4-182, add the following to the **Start** parameter:

The starting index must be greater than or equal to zero.

Also, change CLR_NOINDEX in the description of the **Array** parameter to QLCT_NOTLOADED.

GpiSetClipRegion

On page 4-266, add the following to the Remarks section:

It is the responsibility of the application to free the previous clip region (if any) with GpiDestroyRegion, even if this region was not originally created explicitly by the application.

GpiSetMarkerSet

On page 4-302, add the following to the Remarks section:

If the default marker set is changed (using GpiSetDefAttrs), the initial default marker set cannot be selected with GpiSetMarkerSet.

GpiSetMetaFileBits

On page 4-303, add the following to the Remarks section:

The length of the metafile is increased, if necessary, to accommodate the supplied data. If the supplied data is shorter, the metafile length is not reduced. However, in this case the metafile is still valid, providing the data in it is complete and otherwise correct.

GpiSetPatternSet

On page 4-313, add the following to the Remarks section:

If the default pattern set is changed (using GpiSetDefAttrs), the initial default marker set cannot be selected with GpiSetPatternSet.

Advanced Video Function Calls

VioQueryFonts

On page 8-11, add the following to the Remarks section:

A default font, suitable for use with VIO, is provided automatically for each VIO-supported cell size; default fonts are not enumerated by VioQueryFonts. Valid cell sizes for the device can be found using DevQueryCaps (CAPS_CHAR_WIDTH with CAPS_CHAR_HEIGHT and CAPS_SMALL_CHAR_WIDTH with CAPS_SMALL_CHAR_HEIGHT if the device supports two sizes). Devices that support more than two cell sizes also support the DEVESC_QUERYVIOCELLSIZES escape (see DevEscape).

Window Function Calls

RECT Structure

Change the programming note under the parameters that use the *RECT* structure in each of the following calls:

WinBeginPaint	Page 9-18
WinCalcFrameRect	Page 9-20
WinCopyRect	Page 9-28
WinCreateCursor	Page 9-33
WinDrawBitmap	Page 9-79
WinDrawBorder	Page 9-81
WinDrawText	Page 9-84
WinEqualRect	Page 9-94
WinFillRect	Page 9-96
WinInflateRect	Page 9-119
WinIntersectRect	Page 9-124
WinInvalidateRect	Page 9-125

WinInvertRect	Page 9-127
WinIsRectEmpty	Page 9-129
WinOffsetRect	Page 9-161
WinPtInRect	Page 9-169
WinQueryUpdateRect	Page 9-230
WinQueryWindowRect	Page 9-242
WinScrollWindow	Page 9-263
WinSetRect	Page 9-292
WinSetRectEmpty	Page 9-293
WinSubtractRect	Page 9-318
WinUnionRect	Page 9-327
WinValidateRect	Page 9-331

Change the programming note to:

> The value of each field in this structure must be in the range –32 768 through 32 767. The data type WRECT may also be used, if supported by the language.

WinCreateHeap

On page 9-42, add the following to the programming note:

> This is particularly important for the command line and environment parameters (**argv** and **envp**) passed to the main routine of a C-language application. These parameters are pointers to structures that are allocated using C-language memory allocation functions, and will become invalid if WinCreateHeap is used to create a heap in the automatic data segment. In this case, the application must either process these parameters before creating a heap, or use the DosGetInfoSeg call to access the command line and environment.

WinCreateStdWindow

On page 9-49, change the last sentence of the description of the **Style** parameter to:

> The interpretation of the parameters is affected by the use of all the styles, except for WS_MINIMIZED and WS_MAXIMIZED. These two styles are ignored if they are specified.

WinInitialize

On page 9-120, delete the example given in the **hab** parameter.

WinLoadDlg

On page 9-136, change the first paragraph of the Remarks section to:

> Unless window style WS_VISIBLE is specified for the dialog window in the DIALOG statement within the dialog template, the dialog window is created as an invisible window

WinRegisterClass

On page 9-250, change the first sentence to:

> However, if a private class is registered with the same name as one that already exists, the parameters replace the old class parameters and the return value is TRUE.

WinSetClassMsgInterest

On page 9-272, replace the definition of **SMIM_ALL** with:

SMIM_ALL All messages (except for WM_QUIT if **Control** is SMI_AUTODISPATCH or SMI_NOINTEREST).

Add the following to the Remarks section:

The interest for WM_QUIT may not be set to SMI_AUTODISPATCH using SMIM_ALL, because WM_QUIT is the normal means of terminating an application. It may be set specifically if required.

WinSetMsgInterest

On page 9-284, replace the definition of **SMIM_ALL** with:

SMIM_ALL All messages (except for WM_QUIT if **Control** is SMI_AUTODISPATCH or SMI_NOINTEREST).

Add the following to the Remarks section:

The interest for WM_QUIT may not be set to SMI_AUTODISPATCH using SMIM_ALL, because WM_QUIT is the normal means of terminating an application. It may be set specifically if required.

WinTrackRect

On page 9-323, 9-324, and 9-325, replace the Remarks section with:

Remarks

The WinTrackRect call provides general-purpose pointing-device tracking. It draws a rectangle and enables the user to position the entire rectangle, or size a specific side or corner, as required. The resulting rectangle is then returned to the application, which can use this new information for size and position data. The window manager interface for moving and sizing windows by means of the wide sizing borders uses this call for example.

This call enables the caller to control such limiting values as:

- A maximum and minimum tracking size
- Absolute tracking-position limits
- The tracking rectangle side widths
- A restriction of tracking rectangle movements to a predefined positional grid.

It automatically calls WinLockWindowUpdate to prevent output in the window **hwnd** and its descendants while tracking. When tracking has been completed, output is enabled before this call returns. It also determines which button of the pointing device is depressed at the start of the operation, and only completes the tracking operation when the same button is released.

If the **options** parameter of the *TRACKINFO* structure specified by the TF_SETPOINTERPOS value is included, the pointing-device pointer is positioned at the center of the tracking rectangle. Otherwise, the pointing-device pointer is not moved from its current position and a delta is established between the pointing-device position and the part of the tracking rectangle that it moves (the delta is kept constant).

While moving or sizing with the keyboard interface, the pointing-device pointer is repositioned with the tracking rectangle's new size or position.

While tracking, these keys are active:

Enter Accepts the new position or size.

Left cursor Moves the pointing-device pointer and tracking rectangle left.

If the pointing-device pointer is on the upper or lower edge of the tracking rectangle, the pointer is moved to the top-left or bottom-left corner respectively.

Up cursor Moves the pointing-device pointer and tracking rectangle up.

If the pointing-device pointer is on the left or right edge of the tracking rectangle, the pointer is moved to the top-left or top-right corner respectively.

Right cursor Moves the pointing-device pointer and tracking rectangle right.

If the pointing-device pointer is on the upper or lower edge of the tracking rectangle, the pointer is moved to the top-right or bottom-right corner respectively.

Down cursor Moves the pointing-device pointer and tracking rectangle down.

If the pointing-device pointer is on the left or right edge of the tracking rectangle, the pointer is moved to the bottom-left or bottom-right corner respectively.

Esc Cancels the current tracking operation. The value of the tracking rectangle is undefined on exit.

The pointing device and the keyboard interface can be intermixed. The caller need not include the TF_SETPOINTERPOS value to use the keyboard interface, as this value simply initializes the position of the pointing-device pointer.

If TF_GRID is specified in the *TRACKINFO* structure, the interior of the tracking rectangle is restricted to multiples of the values of the **gridwidth** and **gridheight** parameters. The default values for these are the system font character width and half the system font character height respectively.

Tracking movements using the keyboard arrow keys depend on whether or not TF_GRID is specified in the *TRACKINFO* structure. If not specified, the increments are the values of **xkeyboard** and **ykeyboard**. If specified, the increments are the largest multiples of **gridwidth** and **gridheight** that do not exceed **xkeyboard** and **ykeyboard** respectively. If **gridwidth** exceeds **xkeyboard**, or **gridheight** exceeds **ykeyboard**, the keyboard arrow keys do not cause tracking.

The tracking rectangle is usually logically on top of objects it tracks, so that the user can see the old size and position while tracking the new. Thus, it is possible for a window below the tracking rectangle to be updated while part of the tracking rectangle is above it.

Because the tracking rectangle is drawn in exclusive-OR mode, no window can draw below the tracking rectangle (and thereby obliterate it) without first notifying the tracking code, because unwanted areas of the tracking rectangle can be left behind. If the window doing the drawing is clipped out from the window in which the tracking is occurring, this problem does not arise.

To prevent a window that is currently processing a WM_PAINT message drawing over the tracking rectangle, the tracking rectangle is considered a system-wide resource, only one of which can be in use at any time. If there is a risk of the currently-updating window drawing on the tracking rectangle, the tracking rectangle is removed while that window and its children update, and it is then replaced. This is done during the WinBeginPaint and WinEndPaint calls. If the tracking rectangle overlaps, it is removed in the WinBeginPaint call. In the WinEndPaint call, all the children are updated by means of the WinUpdateWindow call before the tracking rectangle is redrawn.

WinTrackRect has a modal loop within it. The loop has a HK_MSGFILTER hook and a MSGF_TRACK hook code.

Programming Note: It is ensured that the rectangle tracked by this call is within the specified tracking bounds and dimensions. If the rectangle passed is out of these bounds, or it is too large or too small, it is modified to a rectangle that meets these limits.

This call requires the existence of a message queue.

Presentation Manager Programming Reference: Volume 2

Button Control Window Processing

On page 13-1, add the following to the list of button control styles:

This style can be ORed with the BS_AUTORADIOBUTTON style:

BS_NOCURSORSELECT The radio button does not select itself when given the focus as the result of an arrow key or tab key.

Frame Control Window Processing

On page 15-11, add the following sentence to the Default Processing section of WM_FORMATFRAME:

If not processed by the client, the frame control window procedure calculates the size and position of all of the standard frame controls.

List Box Control Window Processing

Delete all information about the following messages:

- LM_QUERYCURSOR message on page 16-7
- LM_SETCURSOR message on pages 16-12 and 16-13
- LM_SETSELECTION message on page 16-15.

These messages are not defined. *******

Multi-Line Entry Field Control Window Processing

WM_CONTROL

On page 18-6, change the following values of the **errinfo** parameter:

EFR_RESIZE should be MLFEFR_RESIZE
EFR_TABSTOP should be MLFEFR_TABSTOP
EFR_FONT should be MLFEFR_FONT
EFR_WORDWRAP should be MLFEFR_WORDWRAP
EFR_TEXT should be MLFEFR_TEXT
ETL_TEXTBYTES should be MLFETL_TEXTBYTES

On page 18-7, change the following values of the **errind** parameter:

EFR_TOOMUCHTEXT should be MLFCPBD_TOOMUCHTEXT
EFR_CLPBDERROR should be MLFCPBD_CLPBDERROR

These values should appear before the line that reads:

For a **notifycode** of MLN_MARGIN

MLM_FORMAT

On page 18-12, change the following values of the **format** parameter:

MLE_CFTEXT should be MLFIE_CFTEXT
MLE_NOTRANS should be MLFIE_NOTRANS
MLE_WINFMT should be MLFIE_WINFMT

All MLE_* constants referred to in this chapter should be MLFIE_* constants.

MLM_QUERYSEL

On page 18-21, change the following values of the **querymode** parameter:

0 should be MLFQS_MINMAXSEL
1 should be MLFQS_MINSEL
2 should be MLFQS_MAXSEL
3 should be MLFQS_ANCHORSEL
4 should be MLFQS_CURSORSEL

These changes also apply to the values given under the **reply** parameter.

MLM_SEARCH

On page 18-26, change the following values of the **style** parameter:

MLSS_CASESENSITIVE should be MLFSEARCH_CASESENSITIVE
MLSS_SELECTMATCH should be MLFSEARCH_SELECTMATCH
MLSS_CHANGEALL should be MLFSEARCH_CHANGEALL

MLM_SETFORMATRECT

On page 18-30, change the following values of the **flags** parameter:

MLESFR_MATCHWINDOW should be MLFFMTRECT_MATCHWINDOW
MLESFR_LIMITHORZ should be MLFFMTRECT_LIMITHORZ
MLESFR_LIMITVERT should be MLFFMTRECT_LIMITVERT

Prompted Entry Field Control Window Processing

On page 19-3, delete references to the LM_QUERYCURSOR, LM_SETCURSOR, and LM_SETSELECTION messages. These messages are not defined.

Help Manager Messages

HM_ACTION_BAR_COMMAND

On page 25-1, this message is now named HM_ACTIONBAR_COMMAND.

Resource Files

On page 26-4, in the Directives section replace all occurrences of "#rcinclude" with "rcinclude."

Code Pages

On page 28-1 insert the following line in the list of supported ASCII code pages:

DeskTop Publishing 1146 1004

Your list should now look like this:

	Char Set	Code Page
Canadian-French	993	863
DeskTop Publishing	**1146**	**1004**
Multilingual	980	850
Nordic	995	865
Portuguese	990	860
U.S. (IBM PC)	919	437

The contents of the code page are shown in the following figure. The title of the figure is **DeskTop Publishing: ASCII Code Page 1004.**

1 → 2 ↓	0-	1-	2-	3-	4-	5-	6-	7-		8-	9-	A-	B-	C-	D-	E-	F-
-0			(SP)	0	@	P	`	p				(RSP)	°	À	Ð	à	ð
-1			!	1	A	Q	a	q			‘	¡	±	Á	Ñ	á	ñ
-2			"	2	B	R	b	r		‚	’	¢	²	Â	Ò	â	ò
-3			#	3	C	S	c	s			“	£	³	Ã	Ó	ã	ó
-4	¯		$	4	D	T	d	t		„	”	¤	´	Ä	Ô	ä	ô
-5	˘		%	5	E	U	e	u		…	•	¥	µ	Å	Õ	å	õ
-6	˙		&	6	F	V	f	v		†	–	¦	¶	Æ	Ö	æ	ö
-7			'	7	G	W	g	w		‡	—	§	·	Ç	×	ç	÷
-8	°		(	8	H	X	h	x		^	~	¨	¸	È	Ø	è	ø
-9			)	9	I	Y	i	y		‰	™	©	¹	É	Ù	é	ù
-A	″		*	:	J	Z	j	z		Š	š	ª	º	Ê	Ú	ê	ú
-B	¸		+	;	K	[	k	{		‹	›	«	»	Ë	Û	ë	û
-C	ˇ		,	<	L	\	l	\|		Œ	œ	¬	¼	Ì	Ü	ì	ü
-D			-	=	M	]	m	}				(SHY)	½	Í	Ý	í	ý
-E			.	>	N	^	n	~				®	¾	Î	Þ	î	þ
-F			/	?	O	_	o				Ÿ	¯	¿	Ï	ß	ï	ÿ

Standard Bit-Map Formats

On pages B-2 and B-3, replace the text with the following:

Bit-Map File Format

Presentation Manager uses the same file format for bit maps, icons, and pointers in resource files. In the following description, "bit map" refers to bit maps, icons, and pointers unless otherwise specified.

Two formats are supported. In the first, a single-size version of the bit map is defined. This is used whatever the target device.

The second format allows multiple versions of the bit map to be defined, one device-independent version, and a number of other device-dependent versions, each intended for use with a particular device. When the bit map is loaded, if there is a device-dependent version available that exactly matches the device size of the current device, then it is used; otherwise the device-independent version is used, and stretched or compressed as necessary to match the current device.

For both formats, the definition consists of two sections. The first section contains general information about the type, dimensions, and other attributes of the resource. The second section contains data describing the pels that make up the bit map(s), and is in the format specified in "Bit-Map Data" on page B-1.

In the multiple-version format, the first section contains an array of BITMAPARRAYFILEHEADER structures. The format of this is as follows:

```
typedef struct _BITMAPARRAYFILEHEADER { /* bafh */
    USHORT            usType;
    ULONG             cbSize;
    ULONG             offNext;
    USHORT            cxDisplay;
    USHORT            cyDisplay;
    BITMAPFILEHEADER  bfh;
} BITMAPARRAYFILEHEADER;
typedef BITMAPARRAYFILEHEADER FAR *PBITMAPARRAYFILEHEADER;
```

The fields in BITMAPARRAYFILEHEADER have these meanings:

usType Type of structure. This is:

BFT_BITMAPARRAY (X'4142' - 'BA' for BITMAPARRAYFILEHEADER)

cbSize Size of the BITMAPARRAYFILEHEADER structure in bytes.

offNext Offset of the next BITMAPARRAYFILEHEADER structure from the start of the file

cxDisplay, cyDisplay Pel dimensions of the device for which this version is intended (eg 640 x 480 for VGA).

The device-independent version must be the first BITMAPARRAYFILEHEADER defined.

In the single-size format, the BITMAPARRAYFILEHEADER structure is not present. The definition consists of one or two BITMAPFILEHEADER structures.

The format of the BITMAPFILEHEADER structure is :

```
typedef struct _BITMAPFILEHEADER { /* bfh */
    USHORT            usType;
    ULONG             cbSize;
    INT               xHotspot;
    INT               yHotspot;
    ULONG             offBits;
    BITMAPINFOHEADER  bmp;
} BITMAPFILEHEADER;
typedef BITMAPFILEHEADER FAR *PBITMAPFILEHEADER;
```

BITMAPINFOHEADER is a standard data type (see above, and also *BITMAPINFOHEADER* on page 2-3.

The fields in BITMAPFILEHEADER have these meanings:

usType Type of resource the file contains. The valid values are:

BFT_BMAP (X'4D42' - 'BM' for bit maps)
BFT_ICON (X'4349' - 'IC' for icons)
BFT_POINTER (X'5450' - 'PT' for pointers).
BFT_COLORICON (X'4943' - 'CI' for color icons).
BFT_COLORPOINTER (X'5043' - 'CP' for color pointers).

cbSize Size of the BITMAPFILEHEADER structure in bytes.

xHotspot, yHotspot Coordinates of the hotspot for icons and pointers. This field is ignored for bit maps.

offBits Offset in bytes to the beginning of the bit-map pel data in the file, from the start of the definition.

For icons and pointers, the **bitmapheight** field in **bmp** is actually twice the pel height of the image that appears on the screen. This is because these types actually contain two full bit-map pel definitions. The first bit-map definition is the XOR mask, which contains invert information (0 = no invert, 1 = invert) for the pointer or icon. The second is the AND mask, which determines whether the pointer or the screen is shown (0 = black/white, 1 = screen/inverse screen).

For color icons or pointers, there are two bit-maps involved: one that is black and white and consists of an AND and an XOR mask, and one that is color that defines the color content.

The **bitmapheight** field in the *BITMAPINFOHEADER* structure for the color bit-map must be the real height, that is, half the value specified for the black and white bit-map. The **bitmapwidth** fields must be the same.

The following table shows how these two bit-maps are used for a color icon or pointer:

XOR	AND	COLOR	
1	1	x	Invert screen
0	0	x	Use color x
0	1	x	Transparency
1	0	x	Use color x

For color icons or pointers, two BITMAPFILEHEADER structures are therefore required:

```
BITMAPFILEHEADER     with usType BFT_COLORICON or BFT_COLORPOINTER
   BITMPAINFOHEADER  (part of BITMAPFILEHEADER)
   Color table
BITMAPFILEHEADER     with same usType
   BITMPAINFOHEADER  (part of BITMAPFILEHEADER)
   Color table
..
bits for one bit-map
..
..
bits for other bit-map
..
```

The usType for the first BITMAPFILEHEADER is either BFT_COLORICON or BFT_COLORPOINTER. This means that a second BITMAPFILEHEADER is present as part of the definition of a color icon or pointer. The first BITMAPFILEHEADER structure contains the information for the black and white AND and XOR masks, while the second BITMAPFILEHEADER structure contains the information for the color part of the pointer or icon.

For the multiple version format, the file is as follows:

```
BITMAPARRAYFILEHEADER    for device-independent version
   BITMAPFILEHEADER      (part of BITMAPARRAYFILEHEADER)
      BITMPAINFOHEADER   (part of BITMAPFILEHEADER)
      Color table

   BITMAPFILEHEADER      )
      BITMAPINFOHEADER   )  only if this is a color icon or pointer
      Color table        )

BITMAPARRAYFILEHEADER    for first device-dependent version
   BITMAPFILEHEADER      (part of BITMAPARRAYFILEHEADER)
      BITMPAINFOHEADER   (part of BITMAPFILEHEADER)
      Color table

   BITMAPFILEHEADER      )
      BITMAPINFOHEADER   )  only if this is a color icon or pointer
      Color table        )

Further BITMAPARRAYFILEHEADER groups occur here as required
for additional device-dependent versions

..
bits for one bit-map
..
..
bits for next bit-map
..

And so on for as many bit-maps as necessary.
```

Presentation Manager C/2 Bindings Reference

Data Types

HLIB

On page 2-10, change the definition to:

```
typedef HMODULE HLIB;
```

MARGSTRUCT

On page 2-12, this data type is now named *MLEMARGSTRUCT*.

Change the line that reads:

```
USHORT   fsFlags;
```

to:

```
USHORT   afFlags;
```

OVERFLOW

On page 2-14, this data type is now named *MLEOVERFLOW*.

MLE_SEARCHDATA

On page 2-13, replace the description of this data type with:

```
typedef struct _MLE_SEARCHDATA {
  USHORT   cb;          /* size of MLE_SEARCHDATA structure */
  PCHAR    pchFind;     /* string to search for */
  PCHAR    pchReplace;  /* string to replace with */
  SHORT    cchFind;     /* length of pchFindString */
  SHORT    cchReplace;  /* length of replace string */
  IPT      iptStart;    /* point at which to start search */
                        /* (negative indicates cursor pt) */
                        /* becomes pt where string found */
  IPT      iptStop;     /* point at which to stop search */
                        /* (negative indicates EOT) */
  USHORT   cchFound;    /* Length of found string at iptStart */
 } MLE_SEARCHDATA;
```

Presentation Manager Macro Assembler/2 Bindings Reference

Data Types

MARGSTRUCT

On page 2-10, this data type is now named *MLEMARGSTRUCT*.

OVERFLOW

On page 2-12, this data type is now named *MLEOVERFLOW*.

Window Function Calls

On page 9-30, add this call:

WinQuerySessionTitle

```
EXTRN WinQuerySessionTitle:FAR
INCL_WINSWITCHLIST  equ 1 ;Or use INCL_WIN or INCL_PM

PUSH   DWORD   hab        ;Anchor-block handle
PUSH   WORD    usSession  ;OS/2 session identity of application whose title is requested
PUSH@  ASCIIZ  szTitle    ;Task title
PUSH   WORD    usTitlelen ;Maximum length of data returnable in bytes
CALL WinQuerySessionTitle

Returns WORD   usRetCode  ;Return code
```

On page 9-44, add this call:

WinSwitchToProgram

```
EXTRN WinSwitchToProgram:FAR
INCL_WINSWITCHLIST  equ 1      ;Or use INCL_WIN or INCL_PM

PUSH  DWORD   hswitchSwHandle  ;Switch list entry handle of program to be activated
CALL WinSwitchToProgram

Returns WORD  usRetCode        ;Return code
```

Presentation Manager COBOL/2 Bindings Reference

Introduction

On page 1-2, replace the Miscellaneous Definitions section with this new section:

Named Constants

Various constants needed by COBOL Presentation Manager applications are provided as named constants in the COBOL header files, PM.CIN and OS2.CIN. Therefore, these must be included at compile time, by means of the COPY statement for example.

In particular, the following definitions are provided to avoid a clash with COBOL reserved words:

```
77 FALSE-VALUE  VALUE IS 0.
77 TRUE-VALUE   VALUE IS 1.
```

Data Types

On page 2-1, add the following note at the beginning of Chapter 2:

> For the data types of the individual structure elements of those data types that map to *HSTRUCT*, refer to Chapter 2 of the *Presentation Manager Programming Reference: Volume One*.

Ranges

Note the changes to the following data types:

COUNT4	Page 2-6	The range should be 0 through 999 999 999.
COUNT4B	Page 2-6	The range should be 0 through 999 999 999.
LENGTH4	Page 2-16	The range should be 0 through 999 999 999.
LONG	Page 2-16	The range should be –999 999 999 through 999 999 999.
PROPERTY4	Page 2-21	Up to 999 999 999 different properties are available.
ULONG	Page 2-27	The range should be 0 through 999 999 999.
WIDTH4	Page 2-28	The range should be –999 999 999 through 999 999 999.

MARGSTRUCT

On page 2-17, this data type is now named *MLEMARGSTRUCT*.

OVERFLOW

On page 2-19, this data type is now named *MLEOVERFLOW*.

Window Function Calls

WinQuerySwitchList (WMQSLI)

On page 8-36, add the following note:

> The **Count** field of the **SwitchEntries** parameter must be initialized to the number of switch entries for which space is available.

Presentation Manager FORTRAN/2 Bindings Reference

Introduction

On page 1-3, replace the Miscellaneous Definitions section with this new section:

Named Constants

Various constants needed by FORTRAN Presentation Manager applications are provided as named constants in the FORTRAN header files, PM.FIN and OS2.FIN. Therefore, these must be included at compile time, by means of the INCLUDE statement for example.

The following definitions are provided in the FORTRAN header file, OS2.FIN:

```
INTEGER*4  FALSE
PARAMETER (FALSE=0)

INTEGER*4  TRUE
PARAMETER (TRUE=1)

INTEGER*4  NULL
PARAMETER (NULL=0)
```

Data Types

On page 2-1, add the following note at the beginning of Chapter 2:

> For the data types of the individual structure elements of those data types that map to *HSTRUCT*, refer to Chapter 2 of the *Presentation Manager Programming Reference: Volume One*.

HPROGARRAY

On page 2-12, delete the lines:

```
C*** DECLAREs for structure components (EQUIVALENCEd onto
HPROGA_BLOCK)
```

MARGSTRUCT

On page 2-14, this data type is now named *MLEMARGSTRUCT*.

OVERFLOW

On page 2-17, this data type is now named *MLEOVERFLOW*.

Profile Function Calls

PrfQueryProgramHandle (PRQPHD)

On page 6-4, change the line that reads:

```
CHARACTER*(HPROGA_LEN)   PROGARRAY
```

to:

```
INTEGER*(HPROGA_LEN)     PROGARRAY
```

Window Function Calls

WinQuerySwitchList (WMQSLI)

On page 8-38, add the following note:

> The **Count** field of the **SwitchEntries** parameter must be initialized to the number of switch entries for which space is available.

I/O Subsystems and Device Support Volume 1: Device Drivers

Device Driver Overview

Device Driver INIT-Time Function Call Summary

On page 2-8, add the following call to the **Device Driver INIT-Time Function Call Summary**:

DosGetInfoSeg Get Address of System Variables Segment

Device Driver Services

DevHlp Services and Function Codes

On page 5-2, in the table of DevHlp Services and Function Codes, change code **38** to **3A**, and add the following services:

```
VideoPause  3C     Suspend/resume video active threads
DispMsg     3D     Display message (for base device drivers)
```

DevHlp Service Modes

On page 5-4, in the table of DevHlp Service Modes, change code **38** to **3A**, and add the following services:

DevHlp Service	Code	Kernel	Interrupt	User	Init
VideoPause	3C	*	*		*
DispMsg	3D				*

On page 5-5, add the DispMsg and VideoPause services to **10. System Services**.

Add the following DispMsg and VideoPause services to Chapter 5:

DispMsg
Display Message

This function is used to display a message from a base device driver on the system console.

Processing

```
MOV  ES,SEG  Msg_Table     ; ES = segment of message table
MOV  DI,OFFSET  Msg_Table  ; DI = offset of message table
XOR  BX,BX                 ; BX = Reserved (must be 0)
MOV  DL,DevHlp_DispMsg
CALL [Device_Help]
```

Results

None.

Remarks

The message is not displayed immediately, but is queued until system initialization has progressed far enough to be able to retrieve the message from the system message file and display it.

The structure of the message table is:

```
WORD     Message ID
WORD     Number of fill-in items
DWORD    Pointer to first fill-in item of ASCIIZ string
DWORD    Pointer to second fill-in item of ASCIIZ string
 .
 .
 .
DWORD    Pointer to last fill-in item of ASCIIZ string
```

If the message displayed is an error message, and *pausenoerror=no* is not contained in config.sys, the user will be prompted to **press any key to continue** after the message is displayed.

VideoPause
Suspend/Resume Video Active Threads

This routine is called by device drivers when a DMA overrun is reported by the controller. This service will start or stop a high priority thread running, which will effectively halt threads using the CPU for video transfers, thus allowing the diskette DMA to complete properly.

Processing

```
MOV  AL,on_or_off          ; VideoPause flag
                           ; = 0     off
                           ; < > 0   on
MOV  DL,DevHlp_VideoPause
CALL [Device_Help]
```

Results

None.

Remarks

This call should be used after a retry has failed. The VideoPause should be turned on just long enough to accomplish the transfer so as not to impair system performance. If multiple device drivers turn on VideoPause, it is not turned off until all device drivers have turned it off.

Device Drivers

Although Presentation Manager does not support keystroke or mouse monitors, an OS/2 Version 1.1 application could register a keyboard or mouse device monitor for the Presentation Manager session. This means an application could intercept keystrokes or mouse clicks associated with the Presentation Manager session through keystroke or mouse monitors. However, OS/2 Version 1.2 does not allow an application to register a keyboard or mouse device monitor for a Presentation Manager session. The following information documents this restriction.

Keystroke Monitors

On page 6-66, replace the second paragraph under **Keystroke Monitors** with the following:

When DosMonReg is issued with keyboard devices, the INDEX parameter indicates the OS/2 session the caller requested the monitor to be registered on. Note the INDEX parameter indicates an OS/2 session from 0 to 15 (see DosGetInfoSeg).

An INDEX parameter value of -1 indicates that the calling thread's session is being requested. If the caller requests either the Presentation Manager session or the DOS mode session, a MONITORS NOT SUPPORTED error code is returned. For both of these environments, the error return code will be generated regardless of the caller's specific usage (explicit/implicit) of the INDEX parameter.

Monitoring Character Device Data Streams

On page 8-2, replace the second paragraph under **Monitoring Character Device Data Streams** with the following:

The flow of data between a device driver and its device is called a data stream. The device driver defines the data streams for its device and associates the data it receives from a device or application with a particular data stream. Device drivers may have more than one data stream model within or between sessions. For example, the keyboard device driver supports three different data stream models for DOS mode, Presentation Manager, and OS/2 mode sessions. Of these three keystroke data streams, only the OS/2 mode data stream supports keystroke monitors.

On page 8-2, replace fifth paragraph under **Monitoring Character Device Data Streams** with the following:

Just as device drivers define their data streams, device drivers determine how character monitors are chained together. The keyboard and mouse device drivers, for example, support a monitor chain for each OS/2 mode session. The printer device driver supports two monitor chains for each physical printer device. See *Mouse Device Driver*, *Console Device Drivers (Screen and Keyboard)*, and *The Printer Device Driver* sections in this book for a description of how these device drivers support device monitors.

Device Monitor Support Limitations

On page 8-3, replace the first paragraph under **Device Monitor Support Limitations** with the following:

Data streams for character devices may be monitored by a character device monitor, only when the character device driver provides device monitor support. Applications, including both OS/2 mode and Presentation Manager applications, may monitor keystrokes and mouse clicks for all OS/2 mode sessions. However, applications including both OS/2 mode and Presentation Manager applications, may not monitor keystrokes and mouse clicks for the Presentation Manager session.

On page 8-4, change figure 8-3 **OS/2 Monitors and PM Applications**, as follows:

On the far right of the diagram, remove the monitor boxes from the data flow to a Presentation Manager application running in a Presentation Manager session.

Remember that applications may no longer intercept keystrokes or mouse clicks associated with the Presentation Manager session through keystroke or mouse monitors.

I/O Subsystems and Device Support Volume 2: Presentation Driver Interface

GreQueryLogColorTable

On page 12-95, change CLR_NOINDEX in the description of the **pArray** parameter to QLCT_NOTLOADED.

GreCreateLogColorTable

On page 12-96, add the following to the options listed under **flOptions**:

LCOL_SYSCOLOR This flag is set when the color data addressed by **pData** is for the system color table. For details see WinSetSysColors in the *Programming Reference*.

GreEscape, DEVESC_GETCP

On page 12-123, delete all information about the DEVESC_GETCP subfunction of GreEscape. This escape code is not used and presentation drivers should not provide support for it.

GreEscape, DEVESC_SETMODE

In the GreEscape section of Chapter 12, add the following description of the DEVESC_SETMODE subfunction:

The DEVESC_SETMODE escape code requests the presentation driver to set a specific mode of the output device.

DEVESC_SETMODE is currently defined only for code-page mode (mode=0). The action taken by the handling routine in the presentation driver depends upon what support the device provides for multiple code pages:

1. If the device has the requested code page built into it, the handling routine issues the controls to select the code page.
2. If the device supports downloaded code pages, the handling routine issues the calls and controls needed to load and select the requested code page.

Note: DEVESC_SETMODE is metafiled and is recorded by the spooler.

Stack Frame

hdc Device-context handle.

lEscape DEVESC_SETMODE

clnCount Number of bytes pointed to by plnData.

plnData Pointer to an ESCMODE structure:

ulMode Identifies the mode that is to be changed. Defined values for ulMode are:

0 Code page.

abModedata The format and meaning of abModedata depends upon the value of ulMode. When ulMode=0, abModedata contains the requested code page number expressed as a USHORT value.

pcOutCount The handling routine should ignore this parameter.

pOutData The handling routine should ignore this parameter.

Return Codes

The handling routine returns:

DEV_OK	Successful.
DEVESC_NOTIMPLEMENTED	This escape code is not implemented.
DEVESC_ERROR	Error.

GreEscape, DEVESC_DRAFTMODE

On page 12-120, add the following to the description of the DEVESC_DRAFTMODE subfunction:

Note that the driver's response to a call to GreDeviceQueryFonts is influenced by the current setting, off or on, of draft mode. If the device supports fonts that cannot be positioned to the nearest pel, the GreDeviceQueryFonts routine in the driver should only return details of these fonts when draft mode is on.

GreDeviceSetAVIOFont

On page 13-9, the name, function number, and parameters of GreDeviceSetAVIOFont have been changed to allow the graphics engine to pass logical font information to the presentation driver. The function name is changed to GreDeviceSetAVIOFont2 and Presentation Manager's graphics engine has been modified so that calls to load or delete an image font for an AVIO presentation space are passed through the dispatch table to the handling routine for GreDeviceSetAVIOFont2. To minimize the impact on existing drivers, the default handling routine for GreDeviceSetAVIOFont2 removes the logical font information and redirects the call through the dispatch table to GreDeviceSetAVIOFont.

The syntax for GreDeviceSetAVIOFont2 is:

```
(BOOL) GreDeviceSetAVIOFont2 (hdc, pFat, pFontDef,
                              lcid, pInstance, lFunction)
```

Differences between GreDeviceSetAVIOFont2 and GreDeviceSetAVIOFont are shown below. See GreDeviceSetAVIOFont on page 13-9 for a full description.

pFat *(PFATTRS)*
Pointer to a font attributes structure for the specified font.

lFunction *(ULONG)*
The symbolic name NGreDeviceSetAVIOFont2 identifies the function number in the low-order word of **lFunction**.

GreDeviceSetDCOrigin

On page 13-22, change the paragraph under *Support* to:

> This function must be supported by presentation drivers for display devices. Drivers for other device types must not include support for GreDeviceSetDCOrigin: these drivers must allow calls to this function to be directed, through the dispatch table, to the default handling routine.

GreSetupDC

On page 14-77, change the paragraph under *Support* to:

> This function is supported by the graphics engine and can be hooked by presentation drivers for display devices. Drivers for other device types must not hook GreSetupDC.

SplQpQueryDt

On page 17-27, change the description of the second parameter (**paszDatatypes**) to:

> Pointer to an array of pointers that address the locations for the returned names. Each location should be an array of 16 characters to accommodate the name and its terminating null.

Additional Information

Using the MODE MONO Command

If you have a configuration that includes an MPA (monochrome/printer adapter) display and a VGA display, or compatible displays, and if you want the MODE MONO command to take effect on the monochrome display, then change the SET VIDEO_DEVICES statement in CONFIG.SYS

```
from:  SET VIDEO_DEVICES=VIO_IBMVGA,VIO_IBMMPA
to:    SET VIDEO_DEVICES=VIO_IBMMPA,VIO_IBMVGA
```

The switch in the ordering of the keyword names illustrated above is only required in a configuration where both display adapters support monochrome mode. The switch is not required for an MPA/EGA configuration.

Online OS/2 DM and DTL Reference

New to the OS/2 online programming library is the *OS/2 DM and DTL Reference*. This online document duplicates information in the *Dialog Manager Guide and Reference* and the *Dialog Tag Language Guide and Reference*. When you install the tools package diskettes, this program installs automatically. An introduction provides instruction on its use.

Online OS/2 Programming Reference

OS/2 provides an online Programming Reference that allows you to view Presentation Manager and control program function calls. The additional information provided in this *Technical Update* is also included in the online reference.

IBM United Kingdom
International Products Limited
PO Box 41, North Harbour
Portsmouth, PO6 3AU
England

PRINTED IN DENMARK BY VANG RASMUSSEN LTD.

